

# DSC FINAL PROJECT

**Sanchal Sunil Dhurve**

**2174236**

## Introduction

The analysis is about the individual has a diabetes or not which is done by conducting the full data analysis and the analysis of the survey on the basis of the data variables using different supervised and the non-supervised algorithms.

## Summary

The Diabetes Prediction Dataset include patient medical and demographic information as well as the state of their diabetes (positive or negative). Features like age, gender, blood glucose level, smoking history, hypertension, heart disease, body mass index (BMI), and HbA1c level are among the features included in the data. Healthcare providers may find this helpful in identifying individuals who may be at risk of diabetes and in creating individualized treatment plans. Additionally, researchers can also utilize the dataset to investigate the associations between different demographic and medical characteristics and the risk of developing diabetes. The objective is to create three models for classification and clustering analysis, then assess each model's accuracy. Principal component analysis will also be used in the study to determine and examine the differences among the three categories.

## Description of the experiment

I'm using the seven data mining approaches in this project. Data collection and integration, cleaning, preprocessing, clustering, classification, exploration, and evaluation. I'm utilizing ggplot for data visualization in order to determine the distributions of the variables. I'm viewing each numerical distribution as I analyze. Identifying the correlation between the variables, after which I use clustering (K Means) and classification (SVM and KNN) in data mining approaches. To determine which model best fits the experiment and which technique yields the best accuracy. Lastly, I'm utilizing data evaluation to examine or display how well the multi-class classification task is doing. To evaluate the effectiveness of any classification model, I utilize ROC curve measurements.

1. Data gathering and integration
2. Data Exploration
3. Data Preprocessing
4. Clustering
5. Classification
6. Data Evaluation

*The variables in the dataset are:*

1. gender: Gender refers to the biological sex of the individual.
2. age: Age is an important factor as diabetes is more commonly diagnosed in older adults. Age ranges from 0-80 in our dataset.
3. hypertension: Hypertension is a medical condition in which the blood pressure in the arteries is persistently elevated.
4. heart\_disease: Heart disease is another medical condition that is associated with an increased risk of developing diabetes.
5. smoking\_history: Smoking history is also considered a risk factor for diabetes and can exacerbate the complications associated with diabetes.
6. bmi: BMI (Body Mass Index) is a measure of body fat based on weight and height.
7. HbA1c\_level: HbA1c (Hemoglobin A1c) level is a measure of a person's average blood sugar level over the past 2-3 months.
8. blood\_glucose\_level: Blood glucose level refers to the amount of glucose in the bloodstream at a given time.
9. diabetes: Diabetes is the target variable being predicted, with values of 1 indicating the presence of diabetes and 0 indicating the absence of diabetes.

*#Loading required libraries*

```
library(ggplot2)
```

```
library(dplyr)
```

```
##
```

```
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
## filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
## intersect, setdiff, setequal, union
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.3.2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.3.2
```

```
library(cluster)
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(rpart)
```

```
library(e1071)
```

```
library(rattle)
```

```
## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
```

```
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
## cov, smooth, var
```

## a. Data gathering and integration

The first part is to get the data you will use. You may use anything that has not been used in an assignment or tutorial. It must have at least 100 data points and must include both numerical and categorical (or ordinal) variables. I recommend keeping this relatively straightforward because data cleaning can take a lot of time if you choose a large, messy dataset. Kaggle (<https://www.kaggle.com/datasets>) and the University of California at Irvine (UCI) (<https://archive.ics.uci.edu/ml/index.php>) maintain collections of datasets, some even telling you if they are good examples for testing specific machine learning techniques. You may also choose to join together more than one dataset, for example to merge data on health outcomes by US state with a dataset on food statistics per state. Merging data is not required and will earn you a bonus point in this step.

```
diabetesDataSet <- read.csv("C:/Users/Dell/OneDrive/Desktop/DePaul/Fundamentals of
DS/Homeworks/diabetes_prediction_dataset.csv")
head(diabetesDataSet)
```

```
## gender age hypertension heart_disease smoking_history bmi HbA1c_level
## 1 Female 80 0 1 never 25.19 6.6
## 2 Female 54 0 0 No Info 27.32 6.6
## 3 Male 28 0 0 never 27.32 5.7
## 4 Female 36 0 0 current 23.45 5.0
## 5 Male 76 1 1 current 20.14 4.8
## 6 Female 20 0 0 never 27.32 6.6
## blood_glucose_level diabetes
## 1 140 0
## 2 80 0
## 3 158 0
## 4 155 0
## 5 155 0
## 6 85 0
```

## b. Data Exploration

Using data exploration to understand what is happening is important throughout the pipeline, and is not limited to this step. However, it is important to use some exploration early on to make sure you understand your data. You must at least consider the distributions of each variable and at least some of the relationships between pairs of variables.

*#Dimension of the data*

```
dim(diabetesDataSet)
```

```
## [1] 100000    9
```

As this data is too large, as it has 100k observations. So I am taking only 3000 observations as a sample amount of data for doing analysis. Where data contains 1500 observations of people having diabetes and people don't have diabetes each.

```
diabetesData <-
```

```
  rbind(diabetesDataSet[diabetesDataSet$diabetes ==  
0,][sample(1:nrow(diabetesDataSet[diabetesDataSet$diabetes == 0, ]), 1500),],  
diabetesDataSet[diabetesDataSet$diabetes ==  
1,][sample(1:nrow(diabetesDataSet[diabetesDataSet$diabetes == 1, ]), 1500),])
```

*#Shuffling the Data*

```
diabetesData <- diabetesData[sample(nrow(diabetesData)), ]
```

*#Dimension of the data set*

```
dim(diabetesData)
```

```
## [1] 3000    9
```

```
head(diabetesData)
```

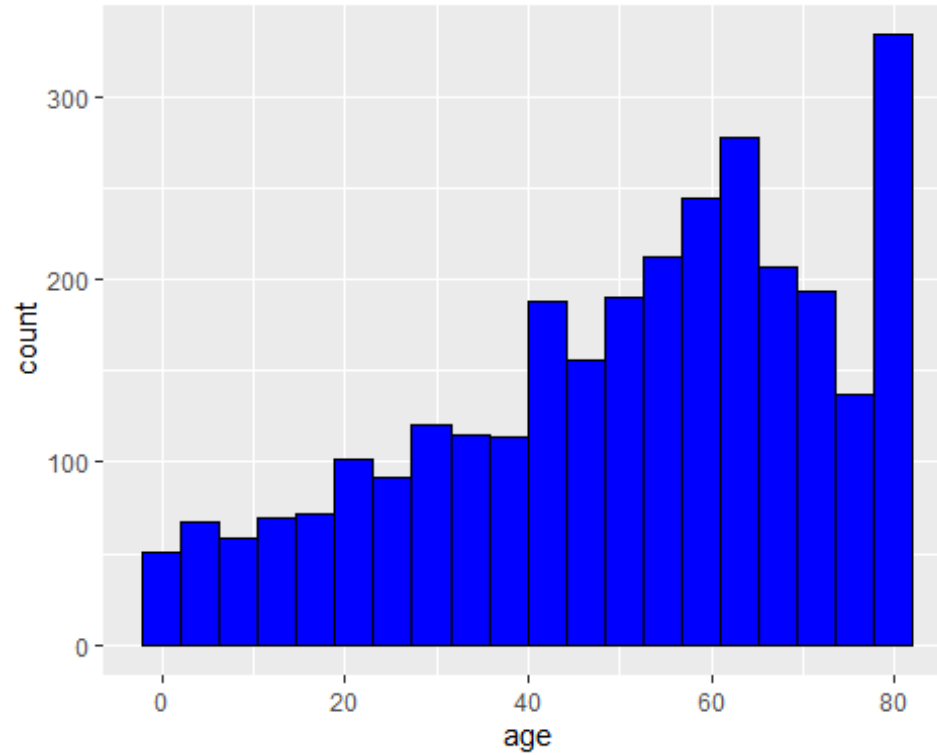
```
##   gender age hypertension heart_disease smoking_history  bmi HbA1c_level  
## 99151 Female 66      0      0      never 36.62      6.6  
## 42865 Female 79      1      0      never 29.05      6.8  
## 2691  Female 32      0      0      No Info 30.86      6.5  
## 69981 Female 51      0      0      never 27.32      4.0  
## 96893 Female 14      0      0      No Info 18.43      6.5  
## 83729 Female 60      0      0      never 25.50      6.1  
##   blood_glucose_level diabetes  
## 99151      155      1  
## 42865      155      1  
## 2691      100      0  
## 69981      90      0  
## 96893      159      1  
## 83729      90      0
```

*#Visualizing the data for variable age*

```
summary(diabetesData$age)
```

```
##   Min. 1st Qu.  Median    Mean 3rd Qu.   Max.  
##  0.16  36.00  54.00  50.61  68.00  80.00
```

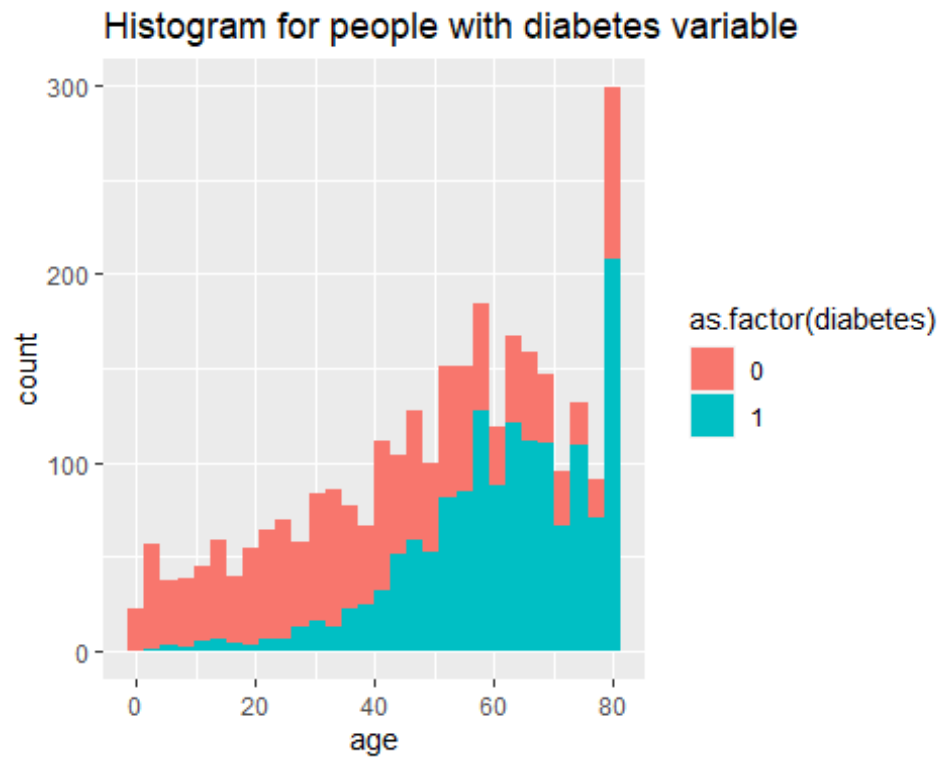
```
ggplot(diabetesData, aes(age)) + geom_histogram(fill = "blue", color = "black", bins = 20)
```



*#Visualizing the data for age with people whether having diabetes or not*

```
ggplot(diabetesData, aes(x = age, fill = as.factor(diabetes))) + geom_histogram() + labs(title = "Histogram for people with diabetes variable")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



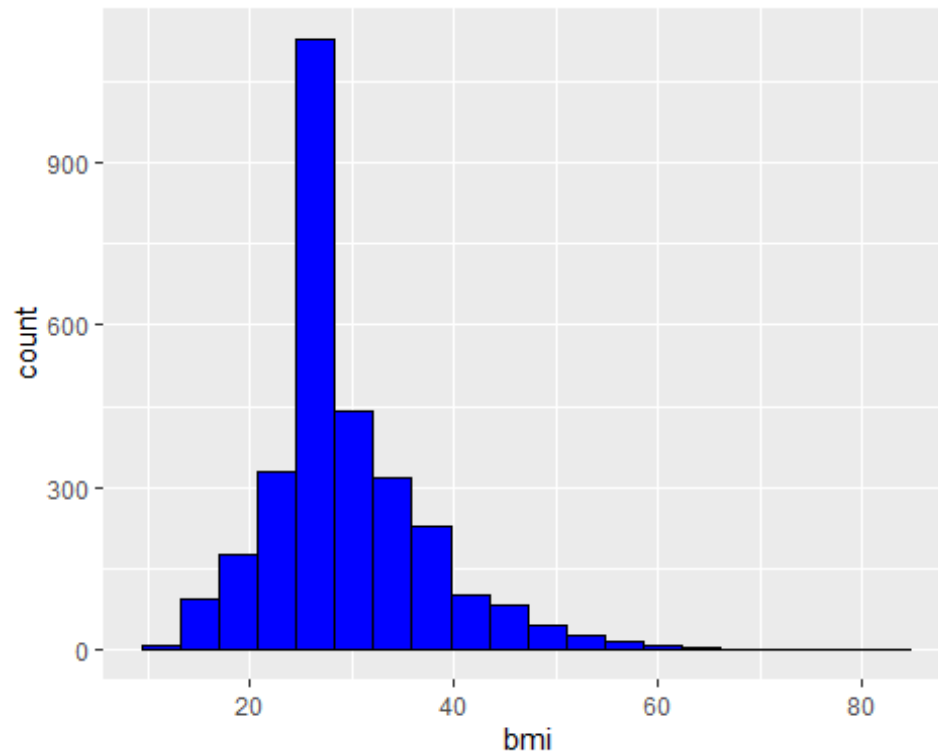
*#Visualizing the data for variable bmi i.e. body mass index*

`summary(diabetesData$bmi)`

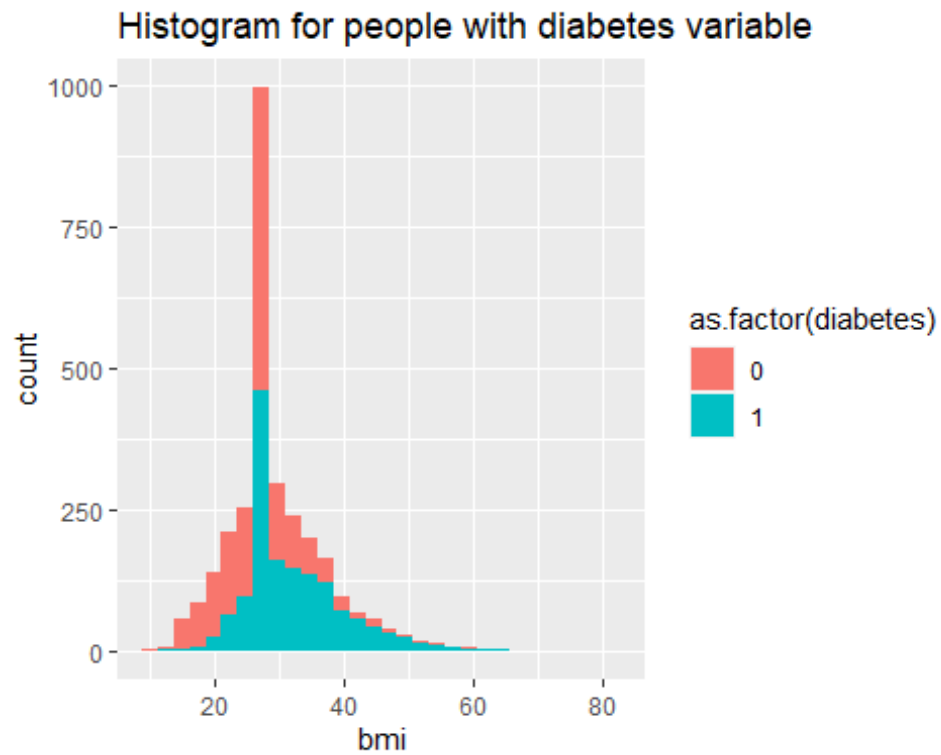
```
##  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
```

```
## 10.01 25.81 27.32 29.51 33.00 81.73
```

```
ggplot(diabetesData, aes(bmi)) + geom_histogram(fill = "blue", color = "black", bins = 20)
```



```
#Visualizing the data for age with people whether having diabetes or not  
ggplot(diabetesData, aes(x = bmi, fill = as.factor(diabetes))) + geom_histogram() + labs(title =  
"Histogram for people with diabetes variable")  
  
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



*#Visualizing the data for variable HbA1c\_level i.e. hemoglobin level*

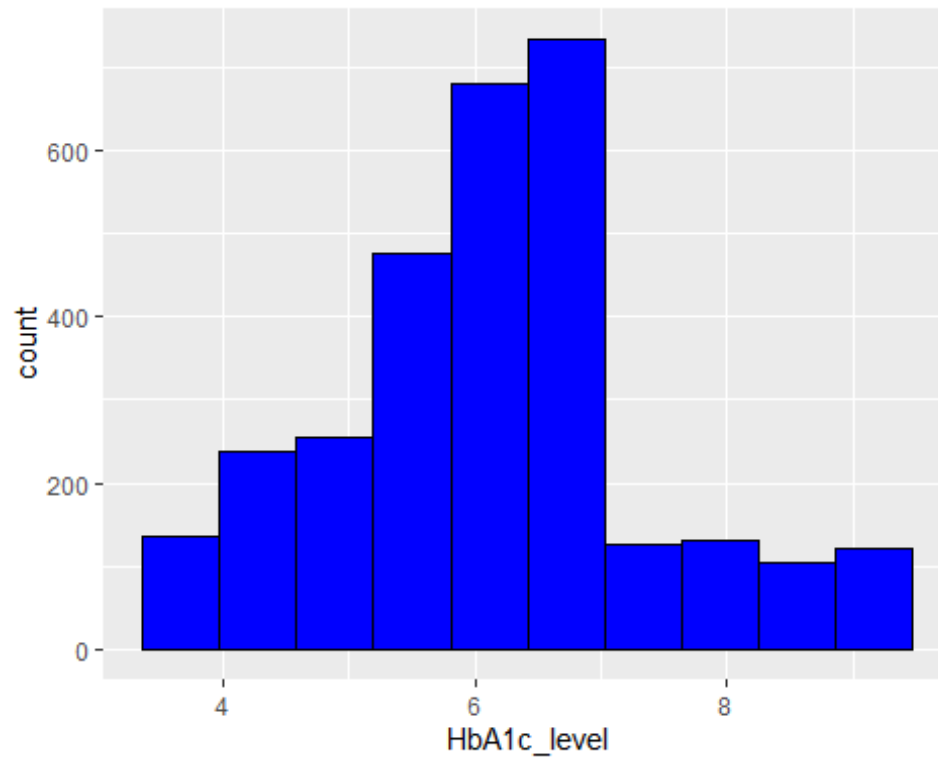
```
summary(diabetesData$HbA1c_level)
```

```
##  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
```

```
## 3.500 5.700 6.100 6.177 6.600 9.000
```

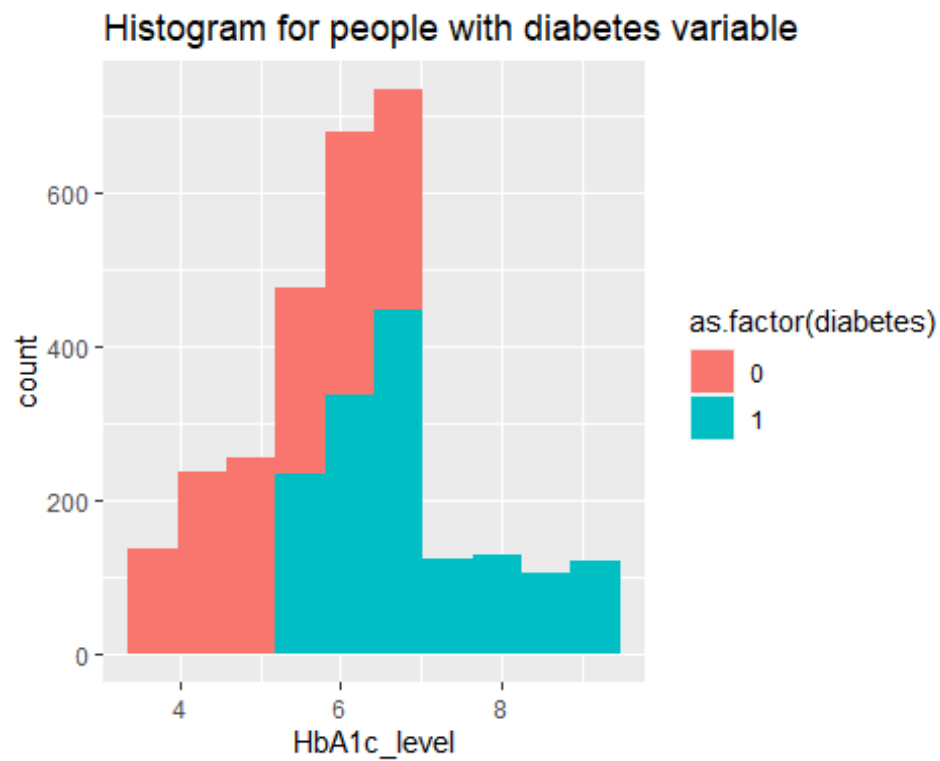
```
ggplot(diabetesData, aes(HbA1c_level)) + geom_histogram(fill = "blue", color = "black", bins = 10)
```





*#Visualizing the data for age with people whether having diabetes or not*

```
ggplot(diabetesData, aes(x = HbA1c_level, fill = as.factor(diabetes))) + geom_histogram(bins = 10) +  
labs(title = "Histogram for people with diabetes variable")
```



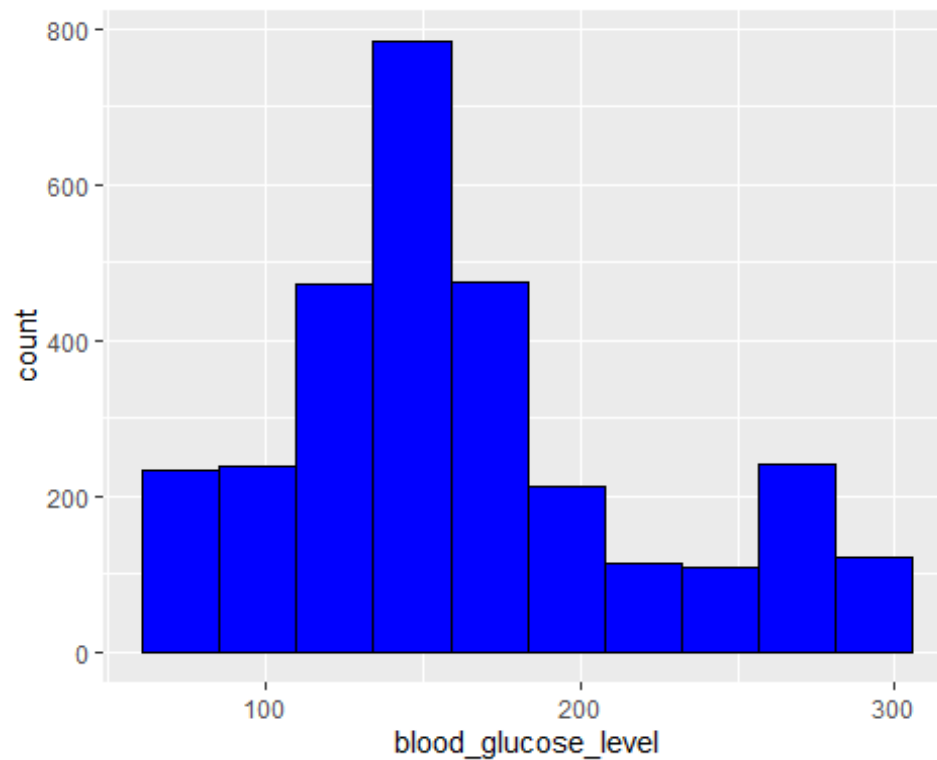
*#Visualizing the data for variable blood\_glucose\_level*

```
summary(diabetesData$blood_glucose_level)
```

```
##  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
```

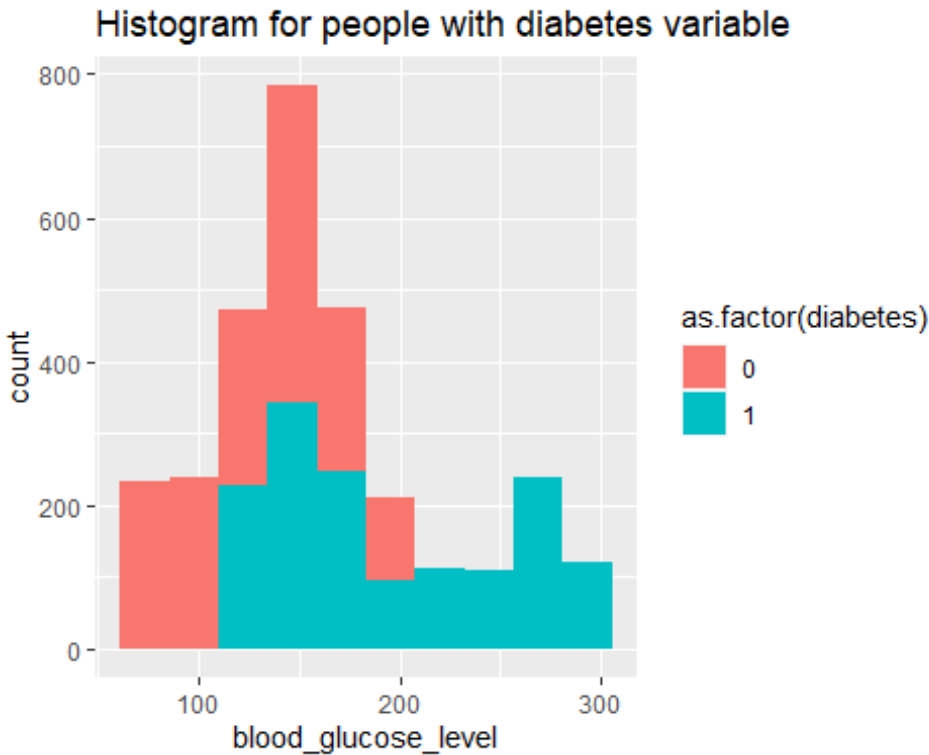
```
##  80.0  130.0  155.0  163.1  200.0  300.0
```

```
ggplot(diabetesData, aes(blood_glucose_level)) + geom_histogram(fill = "blue", color = "black", bins = 10)
```



*#Visualizing the data for age with people whether having diabetes or not*

```
ggplot(diabetesData, aes(x = blood_glucose_level, fill = as.factor(diabetes))) + geom_histogram(bins = 10) + labs(title = "Histogram for people with diabetes variable")
```



There is no need to do barplot for diabetes as there is a data of 1500 observation each.

### c. Data Cleaning

Don't forget – this can take a lot of the time of the whole process. Your cleaning process must ensure that there are no missing values and all outliers must be considered. It may be reasonable to just remove rows with missing values, however, if your data is small or that would change the distributions of the variables, that will not be adequate and you will need to consider other options, as discussed in the modules on cleaning.

The dataset has a variable called `smoking_history` which has a value called `no info` so as it is not needed for the analysis I am converting those values to NA

```
diabetesData$smoking_history <- ifelse(diabetesData$smoking_history == "No Info", NA,
diabetesData$smoking_history)
head(diabetesData)
```

```
##   gender age hypertension heart_disease smoking_history  bmi HbA1c_level
## 99151 Female 66         0           0      never 36.62    6.6
## 42865 Female 79         1           0      never 29.05    6.8
## 2691  Female 32         0           0      <NA> 30.86    6.5
## 69981 Female 51         0           0      never 27.32    4.0
## 96893 Female 14         0           0      <NA> 18.43    6.5
## 83729 Female 60         0           0      never 25.50    6.1
##   blood_glucose_level diabetes
## 99151             155      1
## 42865             155      1
```

```
## 2691      100    0
## 69981      90    0
## 96893     159    1
## 83729      90    0
```

```
summary(diabetesDataSet)
```

```
##  gender      age      hypertension  heart_disease
## Length:100000  Min.   :0.08  Min.   :0.00000  Min.   :0.00000
## Class :character 1st Qu.:24.00 1st Qu.:0.00000 1st Qu.:0.00000
## Mode :character Median :43.00 Median :0.00000 Median :0.00000
##           Mean :41.89 Mean :0.07485 Mean :0.03942
##           3rd Qu.:60.00 3rd Qu.:0.00000 3rd Qu.:0.00000
##           Max. :80.00 Max. :1.00000 Max. :1.00000
## smoking_history  bmi      HbA1c_level  blood_glucose_level
## Length:100000  Min.   :10.01  Min.   :3.500  Min.   :80.0
## Class :character 1st Qu.:23.63 1st Qu.:4.800 1st Qu.:100.0
## Mode :character Median :27.32 Median :5.800 Median :140.0
##           Mean :27.32 Mean :5.528 Mean :138.1
##           3rd Qu.:29.58 3rd Qu.:6.200 3rd Qu.:159.0
##           Max. :95.69 Max. :9.000 Max. :300.0
## diabetes
## Min.   :0.000
## 1st Qu.:0.000
## Median :0.000
## Mean   :0.085
## 3rd Qu.:0.000
## Max.   :1.000
```

As it can be seen in the head the values which are converted to NA are present.

```
#Removing null variables
```

```
sum(is.na(diabetesData))
```

```
## [1] 836
```

```
diabetesData <- na.omit(diabetesData)
```

```
sum(is.na(diabetesData))
```

```
## [1] 0
```

```
dim(diabetesData)
```

```
## [1] 2164  9
```

```
summary(diabetesData)
```

```
##  gender      age      hypertension  heart_disease
## Length:2164   Min.   :0.72  Min.   :0.0000  Min.   :0.0000
## Class :character 1st Qu.:41.00 1st Qu.:0.0000 1st Qu.:0.0000
## Mode :character Median :57.00 Median :0.0000 Median :0.0000
##           Mean :54.08 Mean :0.1895 Mean :0.1026
##           3rd Qu.:69.00 3rd Qu.:0.0000 3rd Qu.:0.0000
```

```
##           Max. :80.00 Max. :1.0000 Max. :1.0000
## smoking_history    bmi      HbA1c_level blood_glucose_level
## Length:2164      Min. :11.95 Min. :3.500 Min. : 80
## Class :character 1st Qu.:26.25 1st Qu.:5.700 1st Qu.:130
## Mode :character Median :28.05 Median :6.200 Median :155
##           Mean :30.50 Mean :6.267 Mean :167
##           3rd Qu.:34.31 3rd Qu.:6.800 3rd Qu.:200
##           Max. :81.73 Max. :9.000 Max. :300
## diabetes
## Min. :0.0000
## 1st Qu.:0.0000
## Median :1.0000
## Mean :0.5689
## 3rd Qu.:1.0000
## Max. :1.0000
```

As per the data set now we have the clean data, can able to proceed with Data preprocessing.

#### d. Data Preprocessing

In some cases, preprocessing is absolutely necessary. It is rarely a bad idea. Make the case for what is and is not necessary given what you plan to do with the data. This could include making dummy variables, applying normalization, binning and/or smoothing, and other transformations (see course module).

*#Creating dummy variables for smoking\_history and gender variable*

```
diabetesData$diabetes <- as.factor(diabetesData$diabetes)
diabetesData$gender <- as.factor(diabetesData$gender)
diabetesData$smoking_history <- as.factor(diabetesData$smoking_history)
diabetesData<- cbind(diabetesData,model.matrix(~ gender -1, data = diabetesData))
diabetesData<- cbind(diabetesData,model.matrix(~ smoking_history -1, data = diabetesData))
head(diabetesData)
```

```
##   gender age hypertension heart_disease smoking_history    bmi HbA1c_level
## 99151 Female 66         0         0      never 36.62      6.6
## 42865 Female 79         1         0      never 29.05      6.8
## 69981 Female 51         0         0      never 27.32      4.0
## 83729 Female 60         0         0      never 25.50      6.1
## 47091 Male 58         0         0      current 37.65      6.1
## 96663 Female 31         0         0      never 35.80      6.6
##   blood_glucose_level diabetes genderFemale genderMale
## 99151             155      1      1      0
## 42865             155      1      1      0
## 69981              90      0      1      0
## 83729              90      0      1      0
## 47091             145      0      0      1
## 96663              80      0      1      0
##   smoking_historycurrent smoking_historyever smoking_historyformer
## 99151              0              0              0
## 42865              0              0              0
## 69981              0              0              0
```

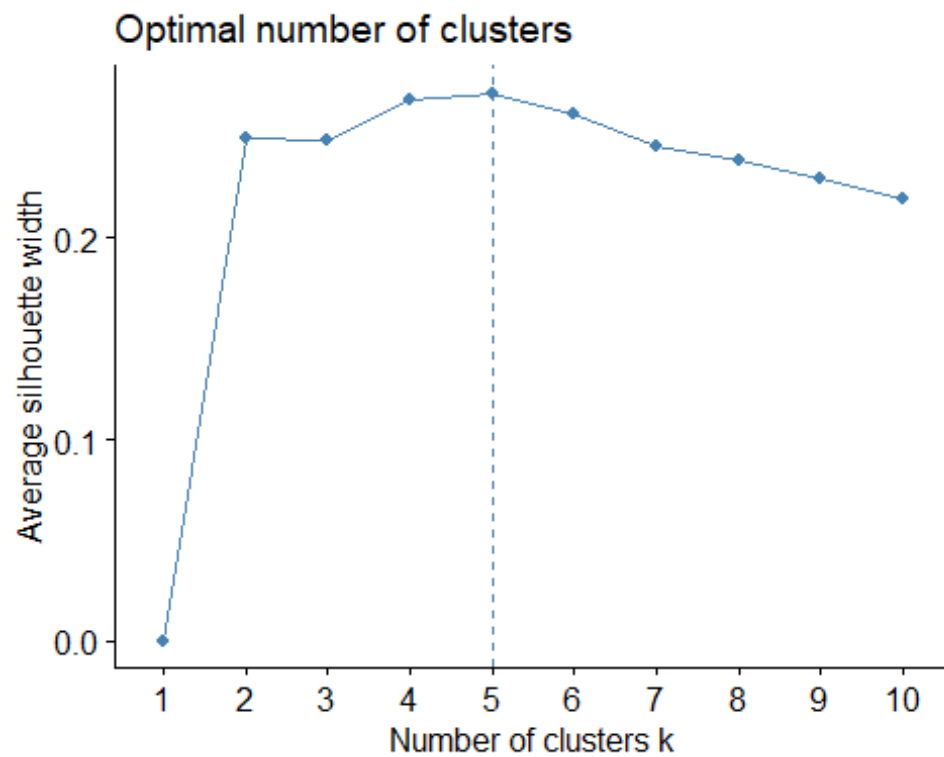
## 83729	0	0	0
## 47091	1	0	0
## 96663	0	0	0
##	smoking_historynever	smoking_historynot	current
## 99151	1	0	
## 42865	1	0	
## 69981	1	0	
## 83729	1	0	
## 47091	0	0	
## 96663	1	0	

As normalization or any other types of transformation or not required for the data set because as it is medical data the values needs to actual and doing binning or smoothing can't be done on the data.

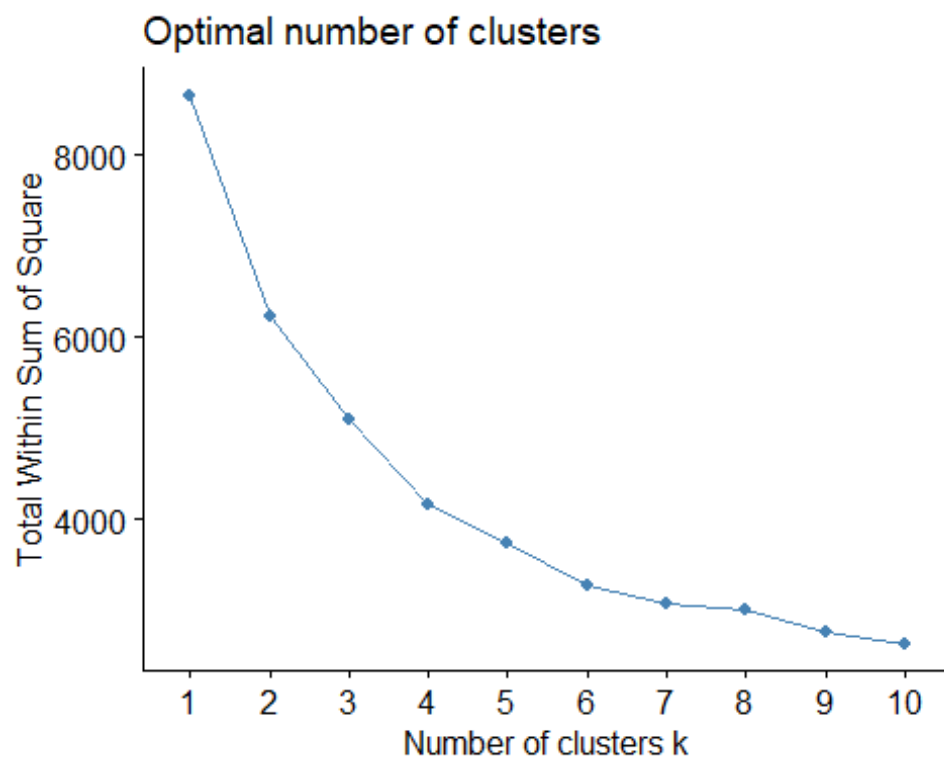
#### d. Clustering

Remove any labels from your data and use clustering to discover any built-in structure. Use an appropriate method to determine the number of clusters. If our data have labels, compare the clusters to those labels. If not, visualize the clustering results by making a PCA projection and coloring the points by cluster assignment. Note that PCA only works for numerical variables, so if your data have just a few categorical, you may skip them. If there are many, use dummy variables or choose a different method for making a projection. One way is to make the distance matrix first (we covered a method for distance matrices using categorical variables in the clustering tutorial) and then apply PCA to that matrix. This is actually a way to calculate an MDS projection, a very popular method.

```
fviz_nbclust(predict(preProcess(diabetesData[,c(2, 6, 7, 8)], method = c("center", "scale")),
diabetesData[,c(2, 6, 7, 8)]),
kmeans, method = "silhouette")
```



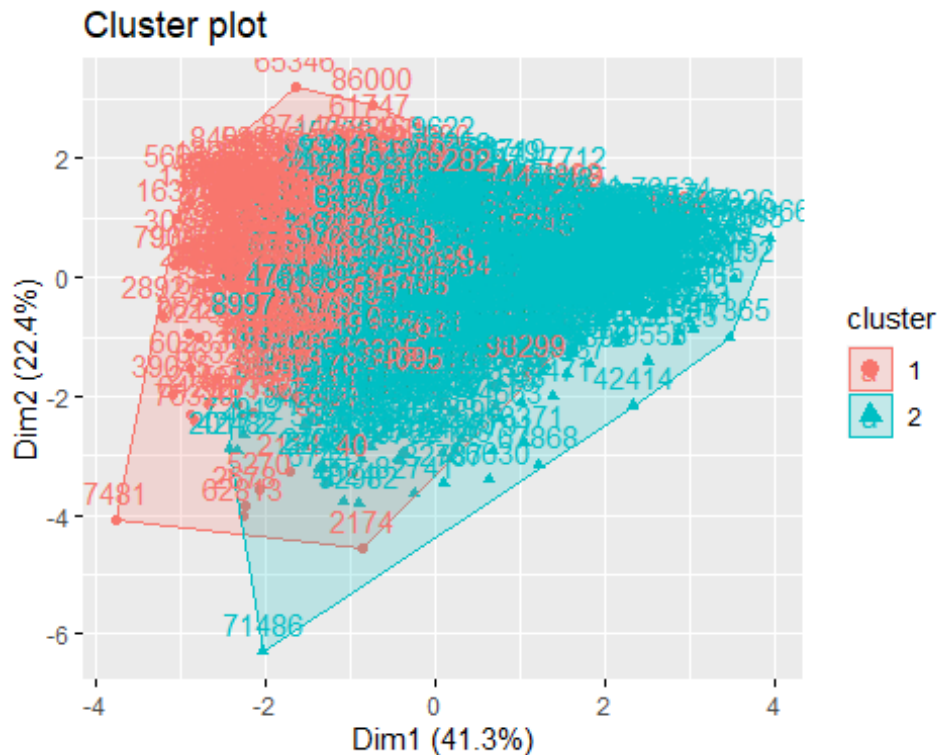
```
fviz_nbclust(predict(preProcess(diabetesData[,c(2, 6, 7, 8)], method = c("center", "scale")),  
diabetesData[,c(2, 6, 7, 8)]),  
kmeans, method = "wss")
```



I am choosing the kneel method to make clusters by that plot k value is 2 as the slight l shape is at 2.

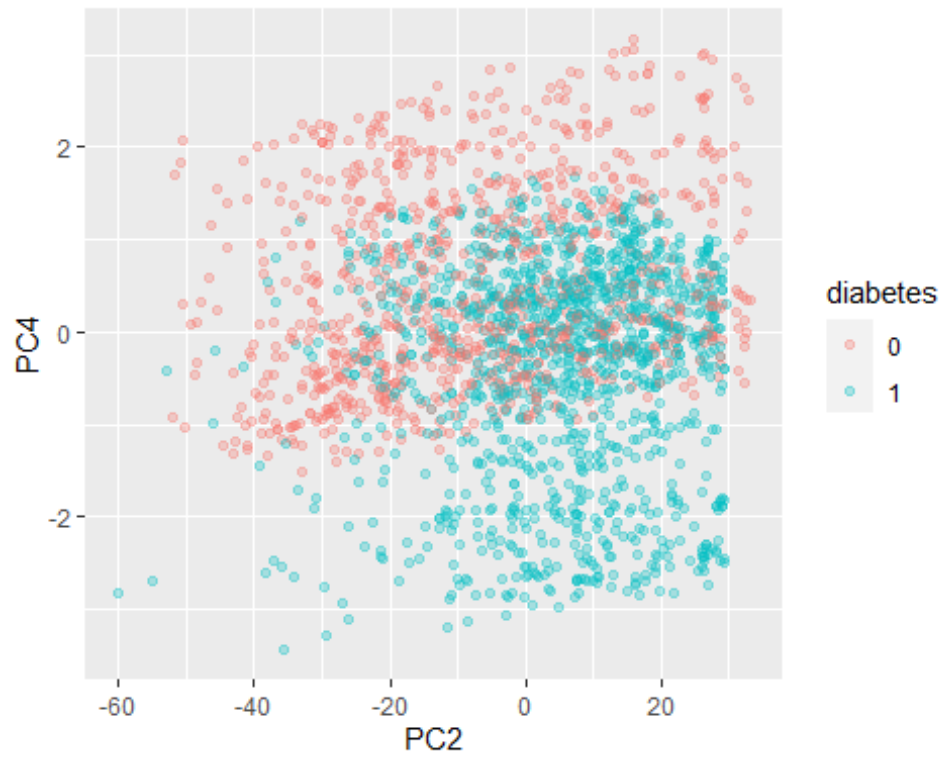
```
kMeansFit <- kmeans(diabetesData[,c(2, 6, 7, 8)], centers = 2, nstart = 25)
```

```
fviz_cluster(kMeansFit, data = diabetesData[,c(2, 6, 7, 8)])
```

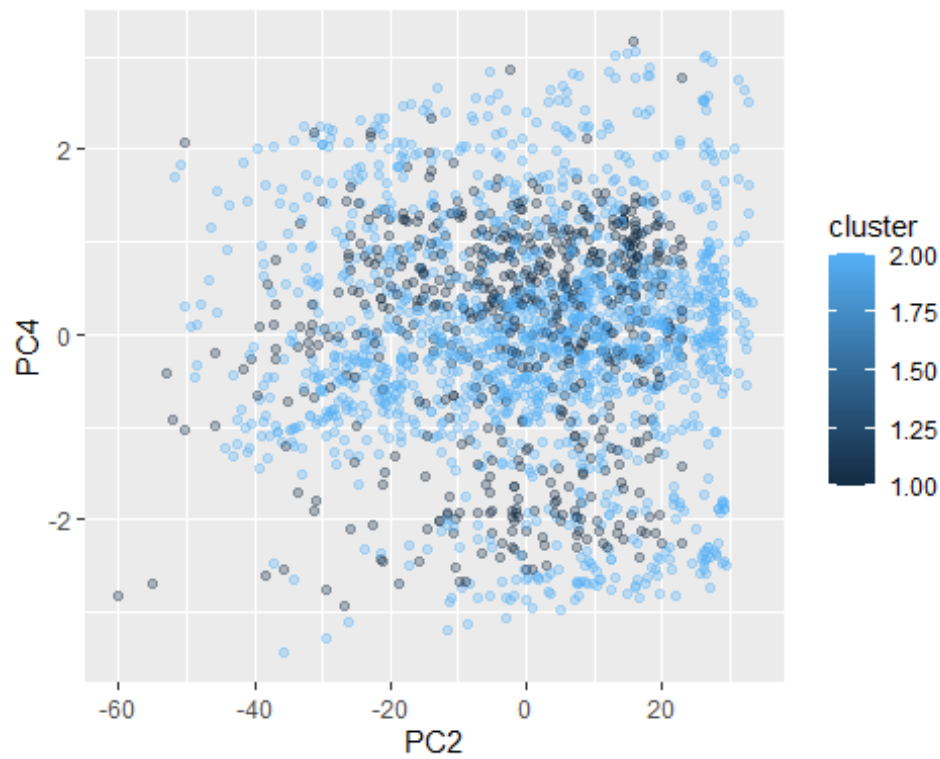


```
pcaDiabetesNumerical <- prcomp(diabetesData[,c(2, 6, 7, 8)])
pcaRotatedDataNumerical <- as.data.frame(pcaDiabetesNumerical$x)
pcaRotatedDataNumerical$diabetes <- diabetesData$diabetes
ggplot(data = pcaRotatedDataNumerical, aes(x = PC2, y = PC4, col = diabetes)) + geom_point(alpha = 0.3)
```





```
pcaRotatedDataNumerical$cluster <- kMeansFit$cluster  
ggplot(data = pcaRotatedDataNumerical, aes(x = PC2, y = PC4, col = cluster)) + geom_point(alpha =  
0.3)
```



As comparing two clusters as per the plot the predicted cluster and actual labels are not same. The clustered model is not good.

## f. Classification

Use at least two classifiers to predict a label in your data. If a label was not provided with the data, use the clustering from the previous part. Follow the process for choosing the best parameters for your choice of classifier. Compare the accuracy of the two.

As I have labels for the data I am not using k means clustering.

### *#Doing PCA for numerical values*

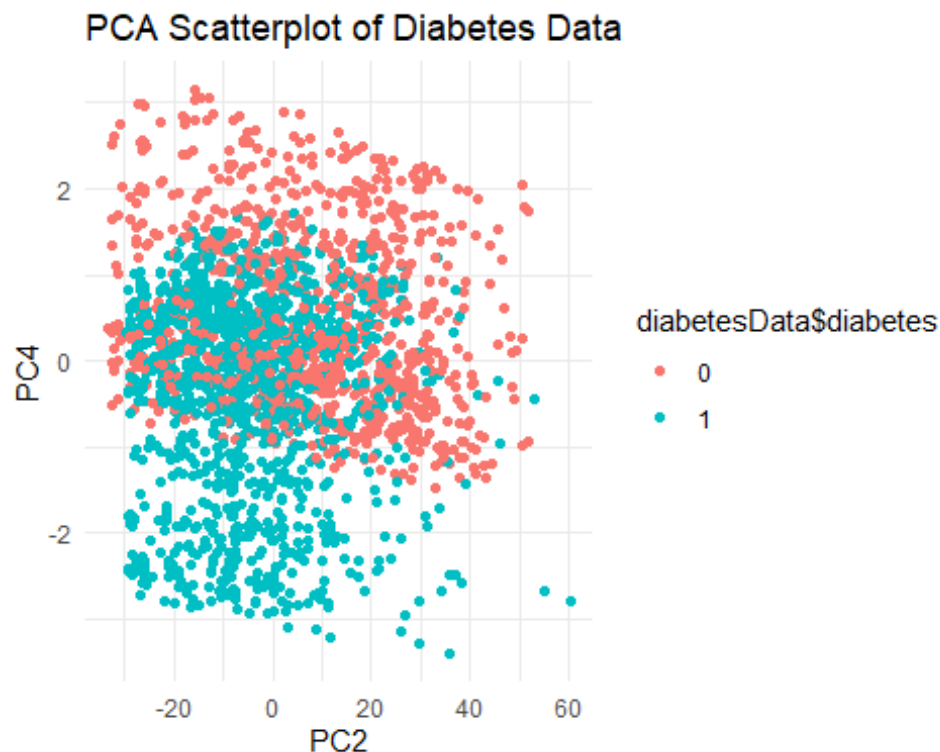
```
diabetesPCADData <- prcomp(diabetesData[, -c(1,5,9)])
str(diabetesPCADData)

## List of 5
## $ sdev   : num [1:13] 58.27 18.18 7.6 1.21 0.71 ...
## $ rotation: num [1:13, 1:13] -0.08419 -0.000805 -0.00049 -0.021445 -0.007236 ...
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : chr [1:13] "age" "hypertension" "heart_disease" "bmi" ...
## .. ..$ : chr [1:13] "PC1" "PC2" "PC3" "PC4" ...
## $ center  : Named num [1:13] 54.079 0.189 0.103 30.495 6.267 ...
## ..- attr(*, "names")= chr [1:13] "age" "hypertension" "heart_disease" "bmi" ...
## $ scale   : logi FALSE
## $ x       : num [1:2164, 1:13] 10.79 9.86 77.03 76.29 21.41 ...
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : chr [1:2164] "99151" "42865" "69981" "83729" ...
## .. ..$ : chr [1:13] "PC1" "PC2" "PC3" "PC4" ...
## - attr(*, "class")= chr "prcomp"

summary(diabetesPCADData)

## Importance of components:
##              PC1  PC2  PC3  PC4  PC5  PC6  PC7
## Standard deviation   58.2702 18.1814 7.60012 1.20666 0.71034 0.54868 0.39438
## Proportion of Variance 0.8967 0.0873 0.01525 0.00038 0.00013 0.00008 0.00004
## Cumulative Proportion 0.8967 0.9840 0.99925 0.99964 0.99977 0.99985 0.99989
##              PC8  PC9  PC10  PC11  PC12  PC13
## Standard deviation   0.37405 0.33044 0.29570 0.27181 7.03e-16 5.067e-16
## Proportion of Variance 0.00004 0.00003 0.00002 0.00002 0.00e+00 0.000e+00
## Cumulative Proportion 0.99993 0.99996 0.99998 1.00000 1.00e+00 1.000e+00

ggplot(data = as.data.frame(diabetesPCADData$x), aes(x = PC2, y = PC4, color =
diabetesData$diabetes)) +
  geom_point() +
  labs(title = "PCA Scatterplot of Diabetes Data") +
  theme_minimal()
```



As per this plot I will use knn for classification, choosing it might be best as per the plot.

```
diabetesKnn <- train(
  diabetes ~ .,
  data = diabetesData[, -c(1, 5)],
  method = "knn",
  trControl = trainControl(method = "cv", number = 10),
  preProcess = c("center", "scale"),
  tuneLength = 15
)
diabetesKnn

## k-Nearest Neighbors
##
## 2164 samples
## 13 predictor
## 2 classes: '0', '1'
##
## Pre-processing: centered (13), scaled (13)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1947, 1948, 1948, 1948, 1948, 1947, ...
## Resampling results across tuning parameters:
##
## k  Accuracy  Kappa
## 5  0.8521271  0.6960005
## 7  0.8548920  0.7016656
## 9  0.8581221  0.7080488
```

```
## 11 0.8558052 0.7034652
## 13 0.8525708 0.6963847
## 15 0.8581413 0.7084586
## 17 0.8595195 0.7114273
## 19 0.8609127 0.7147531
## 21 0.8572133 0.7075301
## 23 0.8585915 0.7105699
## 25 0.8585936 0.7106808
## 27 0.8576741 0.7086214
## 29 0.8544376 0.7020813
## 31 0.8544398 0.7019439
## 33 0.8558222 0.7051493
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 19.
```

*#doing classification using svm*

```
diabetesSvm <- train(
  diabetes ~ .,
  data = diabetesData[, -c(1,5)],
  method = "svmLinear",
  trControl = trainControl(method = "cv", number = 10)
)
diabetesSvm
```

```
## Support Vector Machines with Linear Kernel
##
## 2164 samples
## 13 predictor
## 2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1946, 1948, 1948, 1948, 1948, ...
## Resampling results:
##
## Accuracy Kappa
## 0.8900233 0.775902
##
## Tuning parameter 'C' was held constant at a value of 1
```

As per the classifications done using knn and svm. The svm has highest accuracy of 89% and other two has accuracy of 86% for knn.

## f. Evaluation

Using the better classifier from the previous step, perform a more sophisticated evaluation using the tools of Week 9. Specifically, (1) produce a 2x2 confusion matrix (if your data set has more than two classes, bin the classes into two groups and rebuild the model), (2) calculate the precision and recall manually, and finally (3) produce an ROC plot (see Tutorial 9). Explain how these performance measures makes your classifier look compared to accuracy.

*#creating a test data from the existing dataset*

```
partition <- sample(2, nrow(diabetesDataSet), replace = TRUE, prob = c(0.80,0.20))
```

```
testData <- diabetesDataSet[partition == 2,]
```

```
dim(testData)
```

```
## [1] 19973    9
```

```
testData$diabetes <- as.factor(testData$diabetes)
```

```
testData$gender <- as.factor(testData$gender)
```

```
testData$smoking_history <- as.factor(testData$smoking_history)
```

```
testData<- cbind(testData,model.matrix(~ gender -1, data = testData))
```

```
testData<- cbind(testData,model.matrix(~ smoking_history -1, data = testData))
```

```
predictedDiabetesSvm <- predict(diabetesSvm, testData)
```

```
diabetesConfusionMatrix <- confusionMatrix(testData$diabetes, predictedDiabetesSvm)
```

```
diabetesConfusionMatrix
```

```
## Confusion Matrix and Statistics
```

```
##
```

```
##      Reference
```

```
## Prediction  0    1
```

```
##      0 15999 2208
```

```
##      1   199 1567
```

```
##
```

```
##      Accuracy : 0.8795
```

```
##      95% CI : (0.8749, 0.884)
```

```
## No Information Rate : 0.811
```

```
## P-Value [Acc > NIR] : < 2.2e-16
```

```
##
```

```
##      Kappa : 0.5061
```

```
##
```

```
## McNemar's Test P-Value : < 2.2e-16
```

```
##
```

```
##      Sensitivity : 0.9877
```

```
##      Specificity : 0.4151
```

```
## Pos Pred Value : 0.8787
```

```
## Neg Pred Value : 0.8873
```

```
## Prevalence : 0.8110
```

```
## Detection Rate : 0.8010
```

```
## Detection Prevalence : 0.9116
```

```
## Balanced Accuracy : 0.7014
```

```
##
```

```
## 'Positive' Class : 0
```

```
##
```

*#Confusion matrix for knn*

```
predictedDiabetesKnn <- predict(diabetesKnn, testData)
```

```
diabetesConfusionMatrix <- confusionMatrix(testData$diabetes, predictedDiabetesKnn)
```

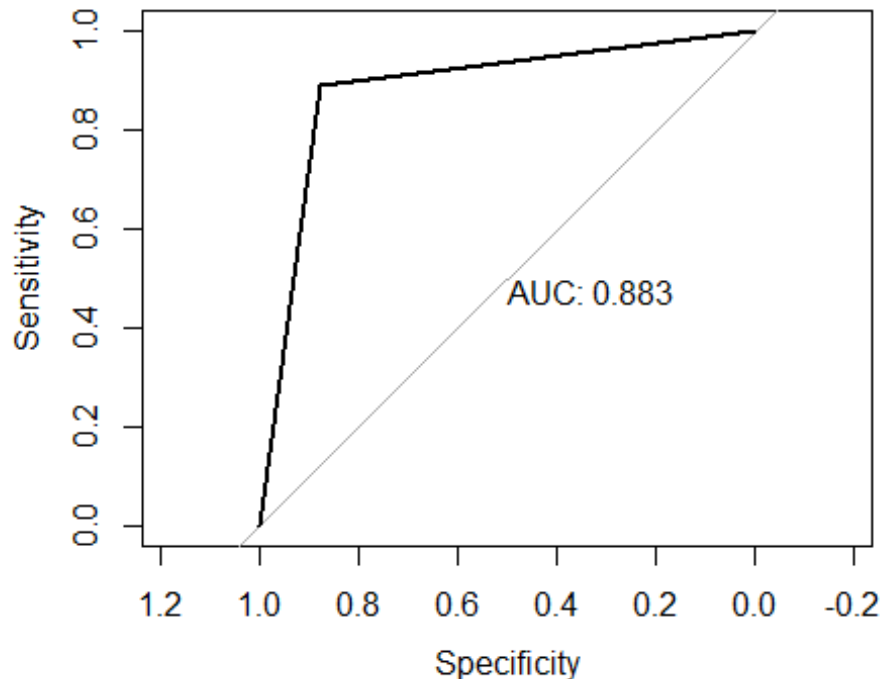
```
diabetesConfusionMatrix
```

```
## Confusion Matrix and Statistics
```

```
##
```

```
##      Reference
```

```
## Prediction    0    1
##           0 15041 3166
##           1   210 1556
##
##           Accuracy : 0.831
##           95% CI : (0.8257, 0.8361)
##           No Information Rate : 0.7636
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.4028
##
##           McNemar's Test P-Value : < 2.2e-16
##
##           Sensitivity : 0.9862
##           Specificity : 0.3295
##           Pos Pred Value : 0.8261
##           Neg Pred Value : 0.8811
##           Prevalence : 0.7636
##           Detection Rate : 0.7531
##           Detection Prevalence : 0.9116
##           Balanced Accuracy : 0.6579
##
##           'Positive' Class : 0
##
roccurve<-roc(response=testData$diabetes, predictor=as.numeric(predictedDiabetesSvm))
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
plot(roccurve, print.auc=TRUE)
```



In this case, the SVM model would be preferred if we wanted a more balanced classifier because it has a greater AUC and balanced accuracy score. Given how little the overall accuracy scores differ, we could conclude that they are rather close. As can be seen from the confusion matrices, the SVM classifier is actually more accurate at predicting the non-dominant class (positives).

#### **h. Report**

After executing the classification methods, I found that the knn is not as accurate as the svm classifier. The reason SVMs were somewhat superior than the decision tree algorithm was because they performed well with generalized data. Furthermore, the kappa number is nearly accurate, suggesting that the data has broader applicability. Due to the two classifiers' extremely high sensitivity, the outcome will not change over time or with successive runs.

#### **i. Reflection**

This course has offered insightful information on a variety of data analysis methods, highlighting the need of comprehending different kinds of datasets, data mining, and data visualization. It emphasizes how crucial preprocessing procedures are, including various cleaning approaches and the use of a variety of methods including binning, smoothing, and normalizing to properly handle missing information. Furthermore, the course explores Predictive Machine Learning, emphasizing the comparison of label values with known data using various "classification" techniques, utilizing SVM, Decision Trees, and KNN parameters. Additionally, mastery of "clustering" was attained, employing k-means models and other techniques to predict label values that are unknown. The course also covered discrimination and class differences in more sophisticated assessment using metrics such as recall, accuracy, and ROC along with information about a model's error rate, proving significant beneficial.